

EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S1	3	tetsuya near tohdo.in.	US-PGPUB; USPAT	OR	ON	2007/05/10 08:20
S2	15	akihito near iwai.in.	US-PGPUB; USPAT	OR	ON	2007/05/10 10:48
S3	8118	denso.as.	US-PGPUB; USPAT	OR	ON	2007/05/10 10:48
S4	9	S3 and (control and simulat\$4 and execut\$4).clm.	US-PGPUB; USPAT	OR	ON	2007/05/10 10:57
S5	1	S3 and (relational adj linkage).clm.	US-PGPUB; USPAT	OR	ON	2007/05/10 11:01
S6	2	("5926638" "6502239").PN.	US-PGPUB; USPAT	OR	ON	2007/05/10 11:01
S7	3855	717/104-105,124-135.ccls.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:07
S8	292	S7 and (control adj program)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:07
S9	143	S8 and model\$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:07
S10	65	S9 and simulat\$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:08
S11	52	S10 and (@pd<"20030228" or @ad<"20030228" or @prad<"20030228" or @rlad<"20030228")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 12:45

EAST Search History

S12	39	adachi near noriyasu.in.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:29
S14	1	wo-200002106-\$.did.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:30
S15	224	vehicle adj control adj program	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:32
S16	9	S15 and simulat\$4 and model\$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:32
S17	161748	control adj (program or application)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:35
S18	205	simulat\$4 with (control adj model)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:35
S19	29	S18 and (control adj program)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:36
S20	895	relationship with model with program	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:45

EAST Search History

S21	74	S20 and "717".clas.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:45
S22	53	S21 and (@pd<"20030228" or @ad<"20030228" or @prad<"20030228" or @rlad<"20030228")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:52
S23	1269	703/21-22.ccls.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:53
S24	16	S23 and (control adj model)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 11:53
S25	920	compiler and break\$1point	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 12:02
S26	306	S25 and model\$4 and simulat\$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 12:03
S27	125	S26 and (simulat\$4 near3 model\$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 12:07
S28	117	S27 and debug\$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 12:08

EAST Search History

S29	108	S28 and (@pd<"20030228" or @ad<"20030228" or @prad<"20030228" or @rlad<"20030228")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 12:08
S30	312	S7 and (model\$4 with simulat\$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 12:36
S31	147	S30 and debug\$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 12:36
S32	30	S31 and break\$1point	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 12:36
S33	43	S31 and breakpoint	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 12:36
S34	165	717/135.ccls.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 12:45
S35	133	S34 and (@pd<"20030228" or @ad<"20030228" or @prad<"20030228" or @rlad<"20030228")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 12:45
S36	7	("5313614" "5577233" "5724590" "5848274" "5943322" "5946474" "6297820").PN.	US-PGPUB; USPAT; USOCR	OR	ON	2007/05/10 14:36

EAST Search History

S37	13	(US-20050138605-\$ or US-20060010429-\$ or US-20030192032-\$).did. or (US-7209800-\$ or US-7085670-\$ or US-7024660-\$ or US-5456604-\$ or US-6052524-\$ or US-6587995-\$ or US-6823497-\$ or US-5923867-\$ or US-6708329-\$).did. or (EP-1111483-\$).did.	US-PGPUB; USPAT; EPO	OR	ON	2007/05/10 17:15
S39	3	S37 and break\$1point	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/10 17:16
S40	15	("20020073380" "20020152060" "5068823" "5566097" "5629876" "5913052" "6023565" "6028993" "6173419" "6321366" "6324672" "6453456" "6460172" "6466898" "6473727").PN.	US-PGPUB; USPAT; USOCR	OR	ON	2007/05/10 17:22
S41	50	("6208799" "4385206" "5454510" "5930142" "4322031" "4377846" "4461958" "4589675" "4785472" "4799145" "4957782" "5007816" "5212430" "5303185" "5332914" "5565895" "5644487" "5661685" "5724565" "5812811" "5874854" "5963563" "5995998" "6018559" "4258425" "4268859" "4305479" "4316390" "4342082" "4347601" "4374420" "4425617" "4458332" "4542452" "4570674" "4577224" "4586151" "4762169" "4791904" "4792996" "4838311" "4839636" "4887296" "4912628" "4916608" "4984151" "5223072" "5230069" "5247234" "5301308").pn.	US-PGPUB; USPAT	OR	ON	2007/05/10 17:30



USPTO

[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide

testing "control program"


[Feedback](#) [Report a problem](#) [Satisfaction survey](#)
Terms used **testing control program**Found **69,775** of **201,062**

Sort results by

relevance

[Save results to a Binder](#)

Display results

expanded form

[Search Tips](#)☐ Open results in a new windowTry an [Advanced Search](#)Try this search in [The ACM Guide](#)

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale ☐ ☐ ☐ ☐ ☐**1** [Second Workshop on Advances in Model-based Software Testing \(A-MOST'06\):](#)[Testing across configurations: implications for combinatorial testing](#)

Myra B. Cohen, Joshua Snyder, Gregg Rothermel

November 2006 **ACM SIGSOFT Software Engineering Notes**, Volume 31 Issue 6**Publisher:** ACM PressFull text available: [pdf\(351.96 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

User configurable software systems allow users to customize functionality at run time. In essence, each such system consists of a family of potentially thousands or millions of program instantiations. Testing methods cannot test all of these configurations, therefore some sampling mechanism must be applied. A common approach to providing such a mechanism has been to use combinatorial interaction testing. To date, however, little work has been done to quantify the effects of different configurati ...

Keywords: code coverage, combinatorial interaction testing, configurable software, empirical study

2 [A technique for testing command and control software](#)

Marvin L. Watkins

April 1982 **Communications of the ACM**, Volume 25 Issue 4**Publisher:** ACM PressFull text available: [pdf\(449.51 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

A technique for testing embedded-microprocessor command and control programs is described. The continuity inherent in functions computed by programs which monitor natural phenomena is exploited by a simple difference equation-based algorithm to predict a program's next output from its preceding ones. The predicted output is compared with the actual output while indexing through the program's domain. Outputs which cannot be predicted are flagged as potential errors. Data are presented which ...

Keywords: predicator

3 [The Test Support Program \(TSP\) a real-time interactive simulation system](#)

Edward G. Ries, Donald J. Harmon

March 1976 **Proceedings of the 1976 ACM SIGMETRICS conference on Computer performance modeling measurement and evaluation SIGMETRICS '76****Publisher:** ACM PressFull text available: [pdf\(659.48 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

The Test Support Program (TSP) is a real-time interactive computer-based simulation model that provides the environment for testing and verifying the operation of netted air control systems. It provides computer-to-computer simulation of data link timing; message interpretation, response, and transmission; and on-line data analysis and recording for up to nine interfaced systems of various types. Event execution and timing is performed in accordance with a time-ordered pre-stored ...

4 Foundations of software testing: dependability theory



Dick Hamlet

December 1994 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 2nd ACM SIGSOFT symposium on Foundations of software engineering SIGSOFT '94**, Volume 19 Issue 5

Publisher: ACM Press

Full text available: [pdf\(1.17 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Testing is potentially the best grounded part of software engineering, since it deals with the well defined situation of a fixed program and a test (a finite collection of input values). However, the fundamental theory of program testing is in disarray. Part of the reason is a confusion of the goals of testing---what makes a test (or testing method) "good." I argue that testing's primary goal should be to measure the dependability of tested software. In support of this goal, a plausible theory o ...

5 Computer aided test pattern generation for digital processors

Klaus Pfeuffer

January 1977 **Proceedings of the 14th conference on Design automation DAC '77**

Publisher: IEEE Press

Full text available: [pdf\(544.66 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Mathematical foundation and algorithm for the implementation of a test pattern generator for digital processors are described.

6 Structured development of graph-grammars for icon manipulation



José Javier Dolado

July 1991 **ACM SIGSOFT Software Engineering Notes**, Volume 16 Issue 3

Publisher: ACM Press

Full text available: [pdf\(602.86 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [index terms](#)

In this work we are showing a structured process to build a grammar for icon manipulation. We presuppose that the object to be manipulated in the computer screen can be stated as a set of relations among its parts. We describe a procedure to generate a program that manipulates the object, guaranteeing that only objects with those properties will be constructed, and that every instance of that object is allowable. The formation rules for the object are stored in terms of attributed graph-grammars ...

7 Extended control program support: VM/370: a hardware assist for the IBM virtual machine facility/370



Arthur G. Olbert

September 1978 **ACM SIGMICRO Newsletter**, Volume 9 Issue 3

Publisher: ACM Press

Full text available: [pdf\(1.34 MB\)](#)

Additional Information: [full citation](#), [abstract](#)


ECPS:VM/370 is a hardware assist of the VM/370 software control program. ECPS:VM/370 provides improved performance for the software system through a combination of hardware assist technologies. The assist provides hardware support for virtual machine execution of certain System/370 real machine functions. The assist also introduces instructions intended for use by the VM/370 control program. ECPS:VM/370 is defined to simplify maintenance and service of the assisted system. This paper describes t ...

8 PODEM-X: An automatic test generation system for VLSI logic structures

Prabhakar Goel, Barry C. Rosales

June 1981 **Proceedings of the 18th conference on Design automation DAC '81**

Publisher: IEEE Press

Full text available:  [pdf\(807.99 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Multiple test generation algorithms and techniques described in this paper have been integrated into a unified system which has successfully produced tests for unpartitioned LSSD logic structures of up to 50,000 logic gates. The design concepts behind the creation of a unified system are presented, as are actual results obtained on large logic structures. System usability was significantly enhanced by the same concepts that facilitated the integration of multiple algorithms and techniques. < ...

9 PODEM-X: An automatic test generation system for VLSI logic structures



P. Goel, B. C. Rosales

June 1988 **Papers on Twenty-five years of electronic design automation 25 years of DAC**

Publisher: ACM Press


Full text available:  [pdf\(1.00 MB\)](#) Additional Information: [full citation](#), [references](#), [index terms](#)

10 Software quality & test accuracy × test coverage

M. Ohba

September 1982 **Proceedings of the 6th international conference on Software engineering ICSE '82**

Publisher: IEEE Computer Society Press

Full text available:  [pdf\(638.27 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The software quality index, SPQL (Software Product Quality Level), is proposed. The index indicates the software quality based on program test results and consists of two subindices: the test accuracy index and the test coverage index. The test accuracy index can be measured by applying the capture-recapture method. Pseudo defects called control defects are seeded prior to the test and their capture ratio is measured as the ratio of the number of detected defects to t ...


11 Projected state machine coverage for software testing



G. Friedman, A. Hartman, K. Nagin, T. Shiran

July 2002 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 2002 ACM SIGSOFT international symposium on Software testing and analysis ISSTA '02, Volume 27 Issue 4**

Publisher: ACM Press

Full text available:  [pdf\(406.24 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Our research deals with test generation for software based on finite state machine (FSM) models of the program specification. We describe a set of coverage criteria and testing constraints for use in the automatic generation of test suites. We also describe the algorithms used to generate test suites based on these coverage criteria, and the implementation of these algorithms as an extension of the Murphi model checker[4]. The coverage criteria are simple but powerful in that they generate te ...


Keywords: automated test generation, finite state machine modeling, state machine projection., validation

12 Using attributed grammars to test designs and implementations

A. G. Duncan, J. S. Hutchison

March 1981 **Proceedings of the 5th international conference on Software engineering ICSE '81**

Publisher: IEEE Press

Full text available:  [pdf\(789.78 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We present a method for generating test cases that can be used throughout the entire life cycle of a program. This method uses attributed translation grammars to generate both inputs and outputs, which can then be used either as is, in order to test the specifications, or in conjunction with automatic test drivers to test an implementation against the specifications. The grammar can generate test cases either randomly or systematically. The attributes are used to guide the genera ...

13 The testing of an APL compiler



Wai-Mee Ching, Alex Katz

September 1993 **ACM SIGAPL APL Quote Quad , Proceedings of the international conference on APL APL '93** , Volume 24 Issue 1

Publisher: ACM Press

Full text available:  [pdf\(797.98 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The testing of the APL-to-C compiler, COMPC, developed at the IBM T.J. Watson Research Center consists of two components: a testing suite of 140 APL programs collected from various sources covering a variety of fields, and a unit-testing procedure which tests each primitive function on all possible subcases arising from different combinations of storage types and shapes. The second component, unit-testing, is an interesting example of the productivity APL can provide for software development. Th ...

14 Artificial intelligence #1: A mobile robot for corridor navigation: a multi-agent approach



Y. Ono, H. Uchiyama, W. Potter

April 2004 **Proceedings of the 42nd annual Southeast regional conference ACM-SE 42**

Publisher: ACM Press

Full text available:  [pdf\(603.53 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

This project focuses on building an autonomous vehicle as the test bed for the future development of an intelligent wheelchair, by proposing a framework for designing and implementing a mobile robot control program that is easily expandable and portable to other robotic platforms. Using a robot equipped with a minimal set of sensors such as a camera and infrared sensors, our multi-agent based control system is built to tackle various problems encountered during corridor navigation. The control s ...

Keywords: collision avoidance, commercial robots and applications, fuzzy logic controller, machine vision, multi-agent systems

15 Evaluating the performance of a unified switching node using a simulated network

Kenneth J. Bodzioch, Bernard E. Patrusky

December 1976 **Proceedings of the 76 Bicentennial conference on Winter simulation WSC '76**

Publisher: Winter Simulation Conference

Full text available:  [pdf\(750.45 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

This paper describes a program which utilizes a discrete event simulation to drive a real switching node. Empirical measurements of various nodal performance characteristics are gathered and recorded during the exercise of this program to aid in the evaluation and design of candidate future nodal architectures. Applications software and specialized hardware for a unified node which switches both digitized voice and data (packet) traffic were developed and tested in a flexible tes ...

16 Generation of hazard free tests using the D-algorithm in a timing accurate system for logic and deductive fault simulation

Eskil Kjelkerud, Owe Thessén

June 1979 **Proceedings of the 16th Conference on Design automation DAC '79**

Publisher: IEEE Press

Full text available:  [pdf\(349.73 KB\)](#) Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)


It is described how a timing accurate system for logic and deductive fault simulation can be used in the forward tracing part of the D-algorithm. The logic simulation is used for the forward implication and the verification phases. The deductive fault simulator is used for D-propagation. Some results from executions of the test generation program are presented.

17 Experimental evaluation of a fuzzy-set based measure of software correctness using program mutation

Farokh B. Bastani, Giuseppe DiMarco, Alberto Pasquini

May 1993 **Proceedings of the 15th international conference on Software Engineering ICSE '93**

Publisher: IEEE Computer Society Press

Full text available:  [pdf\(1.00 MB\)](#) Additional Information: [full citation](#), [references](#)

18 Classics in software engineering

January 1979 Divisible Book

Publisher: Yourdon Press

Full text available:  [pdf\(22.45 MB\)](#) Additional Information: [full citation](#), [cited by](#), [index terms](#)

19 Software/modelware tutorials a: Maximizing simulation ROI with AutoMod: maximizing simulation ROI with AutoMod

Matthew W. Rohrer

December 2003 **Proceedings of the 35th conference on Winter simulation: driving innovation WSC '03**

Publisher: Winter Simulation Conference

Full text available:  [pdf\(525.66 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#)

Simulation modeling and analysis requires an investment in human resources and software. And the rewards from using simulation are significant. Many companies fine tune their operations and reduce waste using simulation. But in the end, every time modeling and analysis are performed, a decision has to be made whether the simulation is "worth doing" (Waite 1999). In this paper we will enumerate how AutoMod has been used to improve return on investment (ROI) from simulation.

20 Taking into account asynchronous signals in functional test of complex circuits

C. Bellon, R. Velazco

June 1984 **Proceedings of the 21st conference on Design automation DAC '84**

Publisher: IEEE Press

Full text available:  [pdf\(625.16 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The proposed functional test method for complex circuits presents the following features:
- the test problem is studied in the aggregate; a test method, a test environment and automated test program generation are proposed. - the circuit behavior is considered as a whole. including the response to instructions (or commands). and to signals at the same level. Emphasis is put on the signal test: an hardware which allows the test of signals and is compatible wi ...

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)